2024W - 192.039 Deep Learning for Natural Language Processing

Reproduction and Extension of
"BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding"

Hunner Markus	Meier Ronja	Steinegger Benno
01503441	12433721	12117772

Contents

1 Introduction

All the code can be found in this GitHub repository: https://github.com/markhun/2024W-DLNLP-BERT. To get things up and running, just use the Makefile and install the `venv` and install the jupyter kernel and you are ready to go to execute the jupyter notebooks.

2 BERT

BERT stands for Bidirectional Encoder Representations from Transformers. It is an encoder-only model which means it only uses the encoder part of the Transformer architecture as shown in 1.

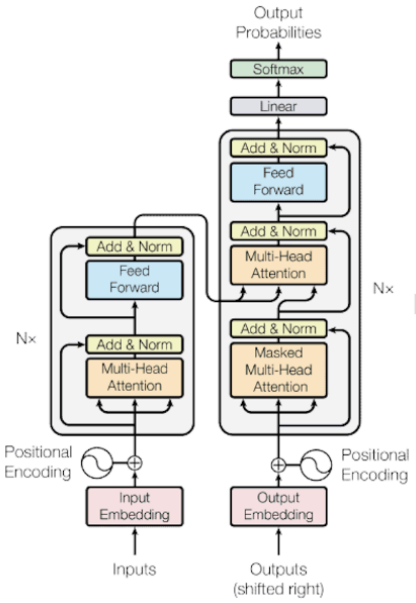


Figure 1: Transformer architecture - Figure 1 in [8]

There is one BERT model with 12 transformer encoder layers called BERT-base and one with 24 called BERT-large. Furthermore, BERT is using bidirectional attention, which lets the model attend to words in both directions to the left and to the right side. This makes it possible for BERT to get a deep understanding of the input text. Therefore, it can excel at Natural Language Processing tasks like text classification or question answering. By using WordPiece Tokenization, the models can store a small vocabulary, which makes the mapping of text to embeddings very efficient and can also cover infrequent terms or newly created words. An example for WordPiece Tokenization is the following: 'unhappiness' is being stored as {'un', '##happiness'}. BERT also uses a couple of special tokens during pretraining and fine-tuning.

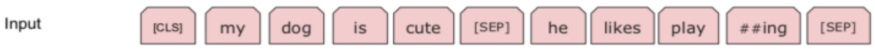


Figure 2: Example for an input sequence of BERT

CLS The classification token is used as a pooling operator to get a single vector per input sequence.

SEP The separator token is used to indicate the end of a sentence in the input sequence.

PAD The padding token is added to sequences to ensure all inputs in a batch have the same length.

MASK The mask token is used in the masked language model to predict this word.

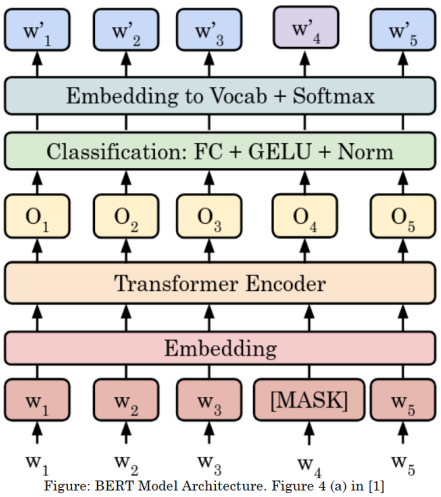


Figure 3: BERT architecture

Figure 2 shows an example for an input sequence and the usage of the special tokens. Figure 3 outlines the architecture of BERT. After the tokens have been embedded they are past on to the transformer encoder layer, which BERT-base has 12 of and BERT-large 24. Every transformer's layer receives as input the output of the previous layer. The green layer symbolizes the model head which can be exchanged depending on the problem. That means if one wants to solve a classification problem the classification head should be attached, if one wants to solve a question-answering task, the question-answering head should be used. Pre-training a large language model requires a huge dataset and needs therefore a lot of computational power. That is why it is very convenient to use a pre-trained model and fine-tune it on an own dataset and for a specific task.

3 ModernBERT

ModernBERT [10] is an update to the well-known and established BERT model. This updated model was released just in December 2024 and still relies on the encoder-only architecture against the current trend towards decoder-only generative models. We want to highlight the metaphor from the ModernBERT paper [10]:

> We've covered this already: encoders are no Ferraris, and ModernBERT is no exception. However, that doesn't mean it can't be fast. When you get on the highway, you generally don't go and trade in your car for a race car, but rather hope that your everyday reliable ride can comfortably hit the speed limit.

3.1 Model Changes

ModernBERT introduces several architectural and training refinements to enhance efficiency and performance. It removes bias terms in all linear layers except the final decoder and disables bias in layer norms. Instead of absolute positional embeddings, it employs Rotary Positional Embeddings (RoPE) for better contextual understanding. Activation functions are updated to GeGLU, a gated-linear variant of BERT's original GeLU, while normalization is improved with a pre-normalization block and standard layer normalization to stabilize training. The Next Sentence Prediction (NSP) task is removed, and Masked Language Modeling (MLM) is expanded to 30% from the traditional 15%, inspired by MosaicBERT. The tokenizer is a BPE-based model from OLMo, retaining the same special tokens as BERT. Structurally, ModernBERT is deeper but narrower, featuring 22 (12 in original BERT) layers for the base model and 28 (24) for the large version, with a significantly higher context length of 8192 tokens. Learning rate and batch size scheduling have been optimized. Attention mechanisms alternate between global and sliding window attention, incorporating Flash Attention 3 for efficiency. Additionally, padding tokens are removed through unpadding, and training is accelerated using `torch.compile`.

4 Recreating Benchmarks

We did generate a lot of figures and plots while running these benchmarks. The most interesting ones can be found in the Appendix 8.

4.1 GLUE

The General Language Understanding Evaluation (GLUE) [9] benchmark is a collection of several natural language understanding tasks, including question answering, sentiment analysis and textual entailment. It consists of nine distinct training datasets, ten validation datasets and eleven test datasets. One of the test datasets (ax) is a diagnostic problem and not part of the benchmark itself. One training dataset (MNLI) provides two validation and test sets, once for entailment and contradiction prediction on in-domain sentence pairs and one for prediction performance on cross-domain sentence pairs. Table 1 shows an overview over all tasks part of the GLUE benchmark.

In [2] BERT is fine-tuned on all GLUE tasks except for WNLI. The authors decided to not include this task within their evaluation, as it contained adversarial examples at the time of publication. For the tasks MRPC and QQP, only the F1 metric is reported. For the STS-B task, only Spearman correlation is reported.

The labels for all GLUE tasks have never been published. The motivation behind this is to keep the ground truth of the benchmark secret, as to prohibit any publication of results which may have used test data for model training. To evaluate a model on the GLUE benchmark suite, the trained model, together with a pre-print of the paper to be published, has to be sent to the University of New York (NYU) for evaluation. NYU maintains an online leaderboard to allow comparison of all models evaluated on the GLUE benchmark.

Therefore, the main problem in reproducing the GLUE results of [2] is to come up with a labeled test dataset that may be somewhat comparable to original testing data provided by GLUE. To achieve this, we prepare a holdout of the provided training data to be used as test dataset. With one exception: For the MNLI task, GLUE provides two test datasets. One only containing in-domain sentence pairs and another one only containing out-of-domain sentence pairs. To produce two comparable testing datasets, we prepare two holdouts of the respective validation datasets. We did not split a test dataset from the provided validation dataset for other

Corpus	|Train|	|Val.|	Task	Metrics	Domain
			Single-Sentence Tasks		
CoLA	8.5k	1k	acceptability	Matthews corr.	misc.
SST-2	67k	872	sentiment	acc.	movie reviews
			Similarity and Paraphrase Tasks		
MRPC	3.7k	408	paraphrase	acc./F1	news
STS-B	7k	1.5k	sentence similarity	Pearson/Spearman corr.	misc.
QQP	364k	40.4k	paraphrase	acc./F1	social QA questions
			Inference Tasks		
MNLI	393k	9.8k/9.8k	NLI	matched acc./mismatched acc.	misc.
QNLI	105k	5.4k	QA/NLI	acc.	Wikipedia
RTE	2.5k	277	NLI	acc.	news, Wikipedia
WNLI	634	71	coreference/NLI	acc.	fiction books

Table 1: Task descriptions and statistics. All tasks are single sentence or sentence pair classification, except STS-B, which is a regression task. MNLI has three classes; all other classification tasks have two. (Table 1 in [9])

tasks, as most tasks only provide very small validation datasets. Preparing a 10 or 20% holdout from such small validation datasets would result in a test dataset far too small to provide any meaningful results. Fortunately, the validation datasets for the MNLI sets consists of close to 10.000 items and are therefore large enough to extract a new test dataset. No data items included in the newly created test dataset were used during model training or validation. Nevertheless, this means that our reproduced results are strictly speaking not comparable to original results of [2], not only because we evaluate our models on a different testing dataset but also because we had to remove the data items held out for testing from the training or validation dataset.

Table 2 shows the evaluation provided in [2] highlighted in yellow. In the original paper, the result for BERT are compared to three models, which were considered state of the art at the time of publication (2018). To put the results in perspective, we added the results for the current top three models listed on the GLUE leaderboard[1] This shows that many of the tasks included within the GLUE benchmark can no longer be considered particularly hard for current state-of-the-art models, with the top three reaching more than 90 points in the respective metrics for most tasks.

To reproduce the results for $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ we fine-tuned the model checkpoints `bert-base-uncased` and `bert-large-uncased` with a learning rate of 2e-5 for five epochs on our modified training data. For evaluation, we choose the best performing epoch checkpoint per GLUE task based on the respective validation dataset performance. As shown in Table 2 this allows us to achieve comparable results to the original paper within all tasks. With the single exception being the QQP task, where our models score more than 18 points above the original results. However, this can anomaly can be attributed to the fact that for the QQP our test dataset is hardly comparable to the original test dataset provided by GLUE (See section 6.1).

4.2 SWAG

The paper "SWAG: A Large-Scale Adversarial Dataset for Grounded Commonsense Inference" [12] introduces SWAG (Situations With Adversarial Generations), a dataset comprising 113,000 multiple-choice questions designed to evaluate a model's ability to predict subsequent events in

[1] See https://gluebenchmark.com/leaderboard - Accessed 2025-01-26.

System	MNLI (m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Avg.**
Current Top 3 Models according to GLUE leaderboard:									
Turing ULR v6	92.5/92.1	76.4	96.7	97.5	73.3	93.1	94.2	93.6	89.9
Vega v1	92.1/91.9	76.7	96.7	**97.9**	**73.8**	93.1	**94.5**	92.4	89.9
Turing NLR v5	**92.6/92.4**	76.4	**97.9**	97.6	72.6	**93.3**	93.8	**94.1**	**90.0**
Models compared to in original paper:									
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
Results of original BERT paper:									
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	86.7/85.9	72.1	92.7	94.9	60.5	86.5	89.3	70.1	82.1
Our reproduction:									
$\text{BERT}_{\text{BASE}}$	86.4/84.2	87.7	89.4	94.6	54.8	84.9	85.6	64.7	81.4
$\text{BERT}_{\text{LARGE}}$	88.2/85.3	88.2	90.5	94.7	58.8	87.6	88.1	67.5	83.2
Our comparison with ModernBERT:									
$\text{ModernBERT}_{\text{BASE}}$	88.4/86.9	88.3	90.6	94.1	58.9	89.0	89.7	76.3	84.7
$\text{ModernBERT}_{\text{LARGE}}$	89.2/87.6	**88.7**	91.5	94.2	60.2	90.6	90.5	83.1	86.2

Table 2: GLUE Test results. The "Average" column is different from the official GLUE score, since the problematic WNLI set was excluded. F1 scores are reported for QQP and MRPC, Spearman correlations are reported for STS-B, Matthew's correlations are reported for CoLA, and accuracy scores are reported for the other tasks. Original results as provided by [2] are highlighted in yellow.

various scenarios. Each question presents a context and four possible continuations, with only one being correct. The correct answer may not be directly entailed from the given input sentence but can be derived from the input in combination with commonsense-knowledge. The data stems from video captions. To mitigate annotation artifacts [4, 6] and human biases, from which the model derives the label from the style instead of the content, the authors propose Adversarial Filtering (AF), a novel method that uses an ensemble of stylistic classifiers to filter out biased data. Empirical results demonstrate that while humans achieve high accuracy (88%) on these questions, models available when the paper was released struggle. Nowadays, models achieve up to 94% [2].

Unfortunately, the gold labels for the `test`-split are not public, therefore we created our own split by merging the train and validation set and then creating train-validate-test split (60%, 20%, 20% respectively) from the merged dataset. This is one of the reasons why we did not get the same results. We used a batch size of 16 for all experiments, a learning rate of 2e-5 and trained for 3 epochs. In the paper, the authors used an older version of GPT, which is trained on BooksCorpus. We use GPT2, which was introduced after the paper has been released. They also mentioned that they added an additional, trainable vector on top, which is passed through a softmax function for the output. We assume that it is the same as in the `AutoModelForMultipleChoice`-Class.

For fine-tuning BERT on the SWAG dataset, input data is first preprocessed by pairing each context sentence (`sent1`) with all four possible endings (`ending0` – `ending3`), forming multiple-choice options. These pairs are tokenized with truncation and a maximum sequence length of 256, then grouped back into sets of four. The `DataCollatorForMultipleChoice` further processes this data by dynamically padding inputs for uniform batch dimensions, reshaping the flattened

[2] See https://leaderboard.allenai.org/swag/submissions/public

inputs back to their original structure, and attaching the corresponding labels as PyTorch tensors. This streamlined pipeline optimizes data handling for multiple-choice classification tasks using Hugging Face's Trainer.

Model	**Validation**		**Test**		**Parameters**
	Paper	**Our**	**Paper**	**Our**	
BERT-Base	81.6	76.6	-	76.1	110M
BERT-Large	86.6	79.3	86.3	80.8	340M
OpenAI GPT	-	66.8	78.0	66.2	117M
ModernBERT-base	-	77.6	-	77.8	149M
ModernBERT-base$_{xavier}$	-	77.7	-	78.0	149M
ModernBERT-large	-	82.0	-	82.1	395M
DeBERTa-v3-base	-	83.6	-	83.4	100M

Table 3: All calculated accuracies for the SWAG-Benchmark and the size of the models. DeBERTa-large and not our finetuned DeBERTa-base [5], is the current leader of the SWAG benchmark with over 94%.

4.3 SQuAD 1.1

The Stanford Question Answering Dataset is a reading comprehension dataset with questions posed by crowdworkers on a set of Wikipedia articles. The training set contains 87599 samples and the validation set 10570. The task is to extract the correct answer from a context to a given question.

System	Dev		Test	
	EM	F1	EM	F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	-	-	82.3	91.2
#1 Ensemble - nlnet	-	-	86.0	91.7
#2 Ensemble - QANet	-	-	84.5	90.5
Published				
BiDAF+ELMo (Single)	-	85.6	-	85.8
R.M. Reader (Ensemble)	81.2	87.9	82.3	88.5
Ours				
BERT$_{BASE}$ (Single)	80.8	88.5	-	-
BERT$_{LARGE}$ (Single)	84.1	90.9	-	-
BERT$_{LARGE}$ (Ensemble)	85.8	91.8	-	-
BERT$_{LARGE}$ (Sgl.+TriviaQA)	**84.2**	**91.1**	**85.1**	**91.8**
BERT$_{LARGE}$ (Ens.+TriviaQA)	**86.2**	**92.2**	**87.4**	**93.2**

Figure 4: SQuAD 1.1 results of the orginal BERT paper [2]

Figure 4 shows the results that were achieved in the BERT paper on the SQuAD 1.1 task. It also compares the results of the paper to other models that were relevant at the time when the paper was released. Since the different pre-training checkpoints that were used for the ensembles have not been published, it was not possible for us to reproduce the ensembles. Furthermore, we ran into a training loss error when training the model on the TriviaQA dataset. The reason for this probably was that the preprocessing of the dataset was done wrong. The SQuAD dataset is in a JSON format, but the TriviaQA dataset not. In addition to that, the TriviaQA dataset is really large. Even when training BERT large on only a split of 50000 training samples it

already takes 17 hours for three epochs. But the fine-tuning of BERT-base and BERT-large on the SQuAD 1.1 dataset worked out well. The results for the Exact-Match and F1 score can be seen in table 4. We got pretty close to all the results in the paper.

Model	EM		F1	
	Paper	Our	Paper	Our
BERT-Base	80.8	79.9	88.5	87.7
BERT-Large	84.1	83.3	90.9	90.5

Table 4: Calculated EM and F1 score for the SQuAD 1.1 task of each model

5 Extensions

5.1 GLUE: Comparing BERT to ModernBERT

As an extension to the evaluation of the original paper we performed the same evaluation on a $\text{ModernBERT}_{\text{BASE}}$ and $\text{ModernBERT}_{\text{LARGE}}$ model for each GLUE task. We used a learning rate of 8e-5 and again choose the best performing epoch checkpoint per GLUE task based on the respective validation dataset performance. As shown in Table 2, $\text{ModernBERT}_{\text{BASE}}$ usually achieves a performance comparable to $\text{BERT}_{\text{LARGE}}$, with the exception of the RTE task where ModernBERT scores roughly 12 to 16 points higher than the original BERT models.

System	MNLI (m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Avg.**
$\text{BERT}_{\text{LARGE}}$	88.2/85.3	88.2	90.5	94.7	58.8	87.6	88.1	67.5	83.2
$\text{ModernBERT}_{\text{BASE}}$	88.4/86.9	88.3	90.6	94.1	58.9	89.0	89.7	76.3	84.7
Speedup (training) $\text{BERT}_{\text{BASE}}$/$\text{ModernBERT}_{\text{BASE}}$	1.38	1.24	1.38	0.99	0.76	1.15	1.18	1.36	1.18

Table 5: GLUE Test results of $\text{BERT}_{\text{LARGE}}$ and $\text{ModernBERT}_{\text{BASE}}$ in relation to achieved training time speedup of $\text{ModernBERT}_{\text{BASE}}$ compared to $\text{BERT}_{\text{BASE}}$.

Not only achieves $\text{ModernBERT}_{\text{BASE}}$ benchmark results comparable to $\text{BERT}_{\text{LARGE}}$, but also achieves a better runtime performance compared to $\text{BERT}_{\text{BASE}}$. Table 5 shows a comparison of the prediction performance between $\text{ModernBERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ and the achieved speedup of the ModernBERT model compared to $\text{BERT}_{\text{BASE}}$ in training time. Only for the CoLA task, ModernBERT is slower than the original BERT model. However, this can be explained by the small size of the CoLA task. Training the $\text{BERT}_{\text{BASE}}$ model on CoLA takes roughly 74 seconds, whereas the ModernBERT variant takes around 97 seconds to train. As ModernBERT consists of more layers and hecen more parameters than the original BERT model, any runtime performance advantage of ModernBERT might be overshadowed by a one-time overhead of initalization for such short training times. For the SST-2 task, runtimes are roughly equal. For all other tasks ModernBERT achieves a speedup between 1.15 to 1.38. This means that the smaller ModernBERT model reaches the prediction performance of BERT's large variant and in addition to way shorter training times than BERT's base variant.

5.2 SNLI-Benchmark

The SNLI dataset [1] contains a huge corpus (570k) of textual entailment examples, consisting of a text and a hypothesis about the text. The task of the model is to judge whether the hypothesis is entailed, contradicted or neutral to the text. We fine-tuned the pre-trained BERT-base model using the SNLI-dataset. Instead of the Huggingface-Trainer class, we used our own training loop in combination with a custom results class to store loss and accuracy during the different training steps. We achieved a test performance of 90.4% using the BERT-base model. We did also create a custom `NLIClassifier`-class instead of using the `AutomodelForSequenceClassification`-class to have the ability to freeze the pre-trained model, which we ended up not doing.

5.3 Multiple-Chioce Class/Head for ModernBERT & GPT-2

As already mentioned in section 4.2, ModernBERT [11] and GPT-2 do not have a MultipleChoice Class/Head. This Python class loads a pre-trained ModernBERT or GPT-2 model, adds dropout for regularization, and includes a classification head to predict the correct answer. The forward function reshapes the input to process all answer choices simultaneously, passes them through ModernBERT/GPT-2, and extracts the hidden state of the first token for ModernBERT and the last token for GPT-2 from each choice. These representations are then passed through the classifier to produce logits for each option, enabling the model to select the most plausible answer. Without this new custom class, the forward function would not be compatible with the structure of the input. In addition, the additional linear classification layer and the softmax function would be missing.

5.4 Randomly initializing the weights

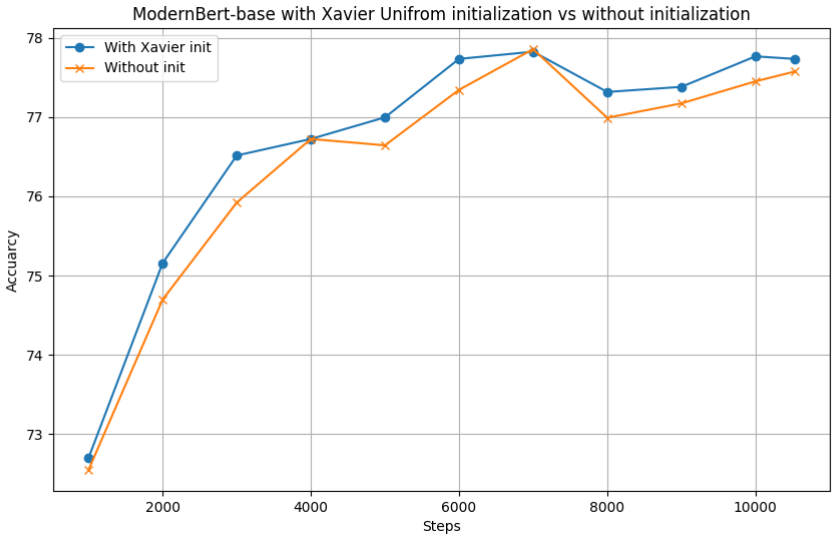


Figure 5: Comparision of validation accuracy between ModernBERT-base with and without weight initialization

Lastly, we want to explore how weight initialization influences the performance of the model. As Dodge et al. [3] and Sutskever et. al. [7] showcase, weight initialization does influence the performance of (deep) neuronal networks. Our experiment is limited to only one layer with 768 (the size of the last hidden layer of ModernBERT) input weights and one output. We assume that the effect would have been greater if we had more weights to initialize. We used the Xavier uniform initialization schema to initialize the weights. As illustrated in the following plot, it did have a (little) positive effect on the accuracy of the model. The final test accuracy with weight initialization was 77.96 % compared to 77.57 % without weight initialization. Due to computing and time restrictions, we limit this experiment to the SWAG benchmark with the ModernBERT-base model.

6 Encountered Issue

6.1 Reproducing GLUE Experiments

The original paper [2] specifies the exact learning rate used to achieve the reported results on the SWAG and SQuAD benchmarks. For GLUE however the paper just lists several numbers and states the "the best fine-tuning learning rate [was selected]" based on results on the validation data.

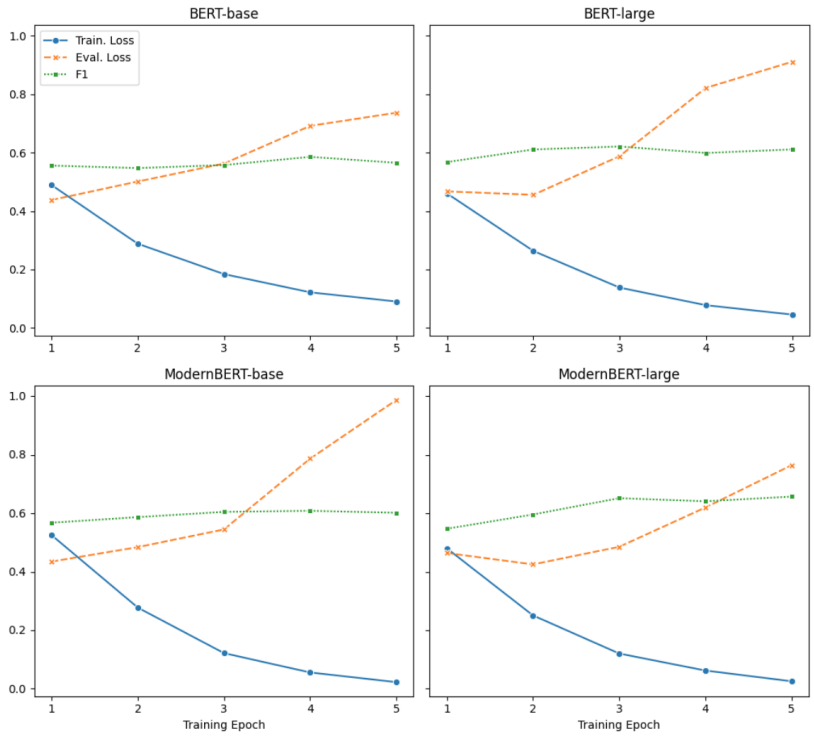


Figure 6: GLUE CoLA fine-tuning - Training loss, Validation loss and F1 score on validation data.

In turn the authors claimed that they fine-tuned for exactly 3 epochs on each GLUE task, however in the appendix section they state that 2, 3 and 4 epochs are considered *good* for most

tasks. We trained for exactly 5 epochs on every GLUE task and then selected the best epoch checkpoint based on performance on the validation dataset. Validation loss plots suggest for some GLUE tasks training for 2 epochs might yield better results than training for 3 or more epochs. As can be seen in Figure 6, validation loss starts to increase very early during the training, suggesting that the model begins to overfit on the training data already after 1 or 2 epochs of fine-tuning.

The original table listing the results for GLUE (Table 1 in [2] lists the training set size of every GLUE task within its second row. Unfortunately, the listed numbers do not match the actual dataset sizes provided by GLUE. For example, QNLI is listed with a training set size of "108k" items. However, the current version of GLUE provides a QNLI training dataset with a little less than 105000 items. In the original paper, the authors refrain from training the WNLI task, as it was considered problematic. In the meantime, GLUE updated the datasets for the WNLI task to resolve this issue. The discrepancy in reported training dataset sizes between the original paper and the current GLUE datasets suggest that also other task datasets were updated. This means that our reproduction of the presented results did not rely on the exact same datasets.

7 Distribution of Work

The following sections were made by Benno Steinegger: SWAG, SNLI-Benchmark, Multiple-Choice Class for ModernBERT & GPT-2, and Randomly initializing the weights. The same split also applies to this report, code and presentation.

The following sections were made by Markus Hunner: 4.1 GLUE, 5.1 GLUE: Comparing BERT to ModernBERT, 6.1 Reproducing GLUE Experiments. The same split also applies to this report, code and presentation.

The following sections were made by Ronja Meier: 2 BERT, 4.3 SQuAD 1.1. The same split also applies to this report, code and presentation.

References

[1] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference, 2015.

[2] Jacob Devlin. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[3] Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*, 2020.

[4] Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. Annotation artifacts in natural language inference data, 2018.

[5] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. DeBERTa: Decoding-enhanced BERT with Disentangled Attention, 2021.

[6] Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. Hypothesis only baselines in natural language inference, 2018.

[7] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1139–1147, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.

[8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[9] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding, 2019.

[10] Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, et al. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. *arXiv preprint arXiv:2412.13663*, 2024.

[11] Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, et al. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. *arXiv preprint arXiv:2412.13663*, 2024.

[12] Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. Swag: A large-scale adversarial dataset for grounded commonsense inference. *arXiv preprint arXiv:1808.05326*, 2018.

8 Appendix

In this part, we want to include some of the most interesting figures which we generated while fine-tuning. All the figures can be found within the git repository.

8.1 GLUE

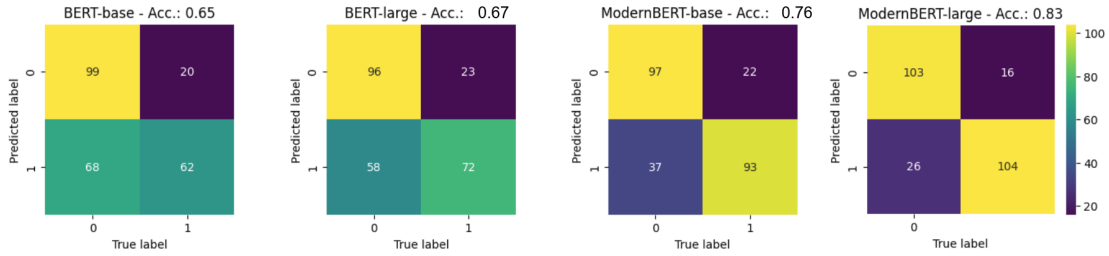


Figure 7: Confusion Matrix for the GLUE RTE task. The original BERT models suffer in accuracy due to mis prediction of the 0 class label. ModernBERT improves on this.

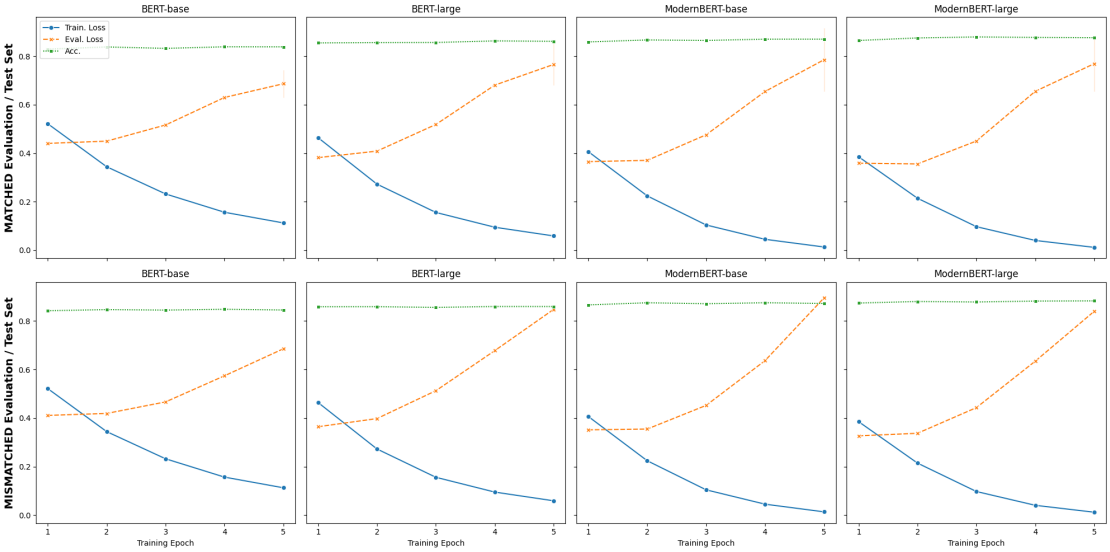
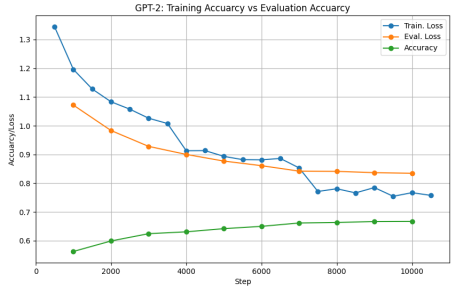
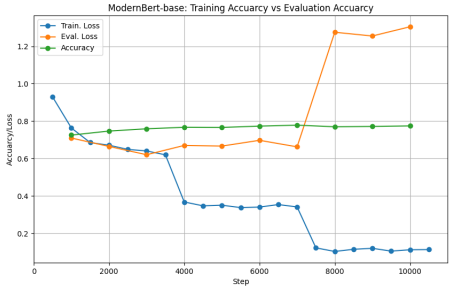


Figure 8: Training and Validation loss and Accuracy for the GLUE MNLI matched/mismatched evaluation. The validation loss curve suggests very early overfitting for this particular GLUE task.

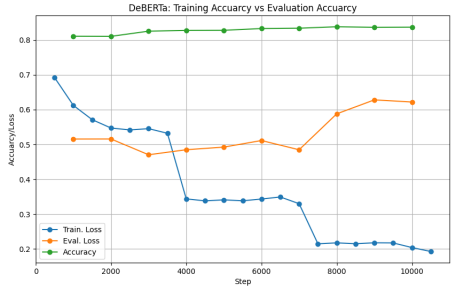
8.2 SWAG



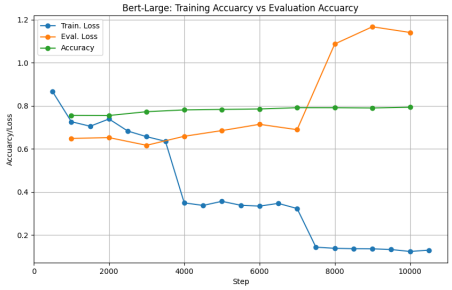
(a) GPT-2 learns without overfitting.



(b) ModernBERT$_{BASE}$ overfits.



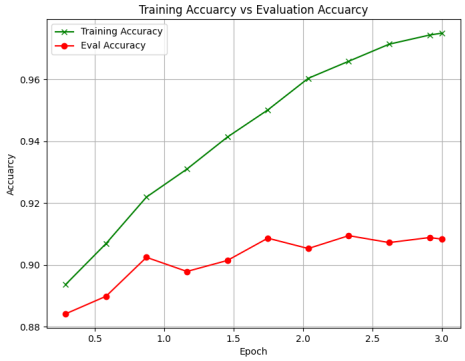
(c) DeBERTa$_{BASE}$ has less loss.



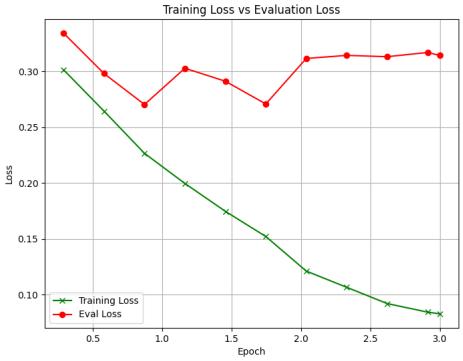
(d) BERT$_{LARGE}$ similar to ModernBERT$_{LARGE}$.

Figure 9: Performance of different models on SWAG dataset.

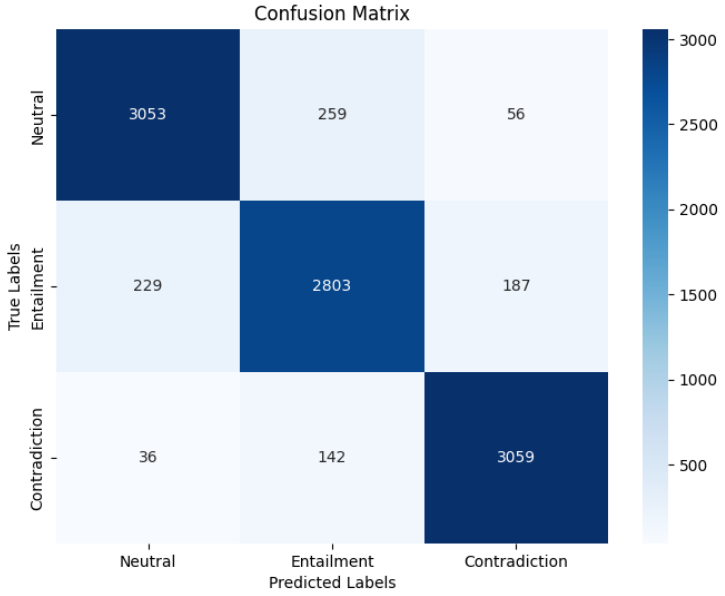
8.3 SNLI



(a) Train vs. val accuracy.



(b) Train vs. val loss.



(c) BERT label classification.

Figure 10: Performance of models on SNLI dataset.